International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 10 – No.5, February 2016 – www.ijais.org

Enhanced Security in Authentication for Web Applications using Anti-Form Grabbing Technique with Email Verification

O.O. Anifowose
Department of Mathematics
Ahmadu Bello University Zaria, Nigeria

S.E. Abdullahi
Department of Mathematics
Ahmadu Bello University Zaria, Nigeria

S.B. Junaidu
Department of Mathematics
Ahmadu Bello University Zaria, Nigeria

ABSTRACT

Online users now make use of internet banking as a major platform of making payments of products online. Cybercriminals are using newer and more advanced methods to target online users. Attacks ranging from reverse social engineering called Phishing; whereby spam messages are sent to customers' emails consisting of links then to fake websites to Trojans that are installed in the user's computer system without his knowledge which monitors the customer's financial transactions on his account which are later used for financial gain which ultimately results in loss of financial funds for affected victims. One of the fastest growing threats and attacks in the world today is Man-in-the-Browser (MITB) attacks. MITB attacks are a specialized and upgraded version of Man-in-the-Middle (MITM) attack whereby it installs itself on the user's PC without the user's concept through internet usage then steals data authentication details and alters legitimate user transactions for the attacker's benefit. As the advance in technology continues to influence the way society makes payment for goods and services, then more advanced security approach is required for transaction verification on the internet.

This paper proposes a more secure authentication for online transaction using Anti-Form Grabbing technique with Email Verification service.

Keywords

Internet banking, Man-in-the-Middle attack, Man-in-the-Browser attack and authentication.

1. INTRODUCTION

When using services in a web environment, security is of great importance. This implies both for the user and the provider. The information in use must be handled in a way that does not compromise its security. Passwords are only secure as long as the user keeps them secret. Not everyone is aware of the risk that comes with compromised passwords and other security leaks.

Lately, client side attacks on online banking and electronic commerce are on the rise due to inadequate security awareness amongst end users. As a result, end user would not be aware if there is vulnerability on their machine or platform that might lead to client side attack such as man-in-the-browser (MitB) attacks. For instance, man-in-the-middle (MitM) attack techniques which are mainly targeting the information flow between a client and a server have now evolved to become man-in-the-browser (MitB) attack. MitM attack occurs when someone manages to eavesdrop on web traffic by fooling the other connections (Web Server and Client Server) to connect to the attacker instead of connecting to each other. One of the common ways to counter these attacks is to use secure channel like SSL when sensitive data is transmitted between the client and the server. MitB attack is designed to infiltrate the client software such as the internet browser and manipulate or steal any sensitive information. It takes place on the client side of the connection. The ability of these trojans to perform Man-In-The-Browser attacks on valid transactions is most worrying since they silently change the information of the user's bank details to the attacker's account details.

The password remains the most popular authentication mechanism in use today. In order to complete any web-based transaction exchange the user will be required to remember and enter their password into an online system.

As technological advances continue to influence the way society makes payment for goods and services, the requirement for more advanced security approaches for transaction verification in the online environment increases.

In order to mitigate these security issues, this paper proffers a solution to the problem by introducing Anti-Form Grabbing which disallows the attacker from "grabbing" sensitive information when they are being sent to the server by the client and also intensify the security by rerouting the verification from the server through Email verification service.

2. RELATED WORKS

In the past, several protection mechanisms have been proposed to prevent man-in-the-browser attacks.

The IBM research team (2008) introduced external device approach to prevent man-in-the-middle (MitM) variant attacks called **Zone Trusted Information Channel (ZTIC)**. The

ZTIC is a USB device containing components such as simple verification display and other authentication components in order to provide secure communication between a client and a server. In this approach, ZTIC acts as a main medium between the client and the server without relying on any client's application or browser. ZTIC will scan and intercept any sensitive information and will only permit to exchange the information once the client has verified the information within its display component. However, the external devices requirement in the above solutions may become barrier for some users.

Wells *et al* (2008) proposed a design which adds another layer of security to existing methods to either prevent a MitM attack or to make the procedure of capturing and reassembling customer log on and transaction details more computationally and time intensive than what it is worth to an attacker. The model is based on a graphical authentication application previously developed called **Authentigraph** for preventing MitM attacks in authentication, data entry and transaction verification. When a client requests an authentication session with the server, an image is generated based on the associated process parameters which are a collection of characters randomly placed within the image. Each character is positioned randomly within its designated quadrant each time an image is generated.

Abbasi *et al* (2010) has proposed a secure web contents to mitigate the MitM and the

MitB attacks. In their solutions, the web contents from a server will be encrypted by a secured web server and a secured proxy on the client side will decrypt the web content. Thus, this solution provides better protection of web content stored in the server. However, this solution concentrates only on the protection of a web contents on the server side and is lacking of protection against the malicious software attack on the client side.

Sidheeq *et al* (2010) integrated the biometrics USB with the TPM(Trust Platform Module) in order to mitigate the risk of malware attacks on the client's machine. In detail, this solution requires the user to provide biometrics evidence for authentication. Then it compares the authenticity of the provided evidence with the user's biometrics stored in the USB which is protected by the TPM. In addition, this solution uses the encryption feature provided by the TPM in order to secure the data exchanged between a client and a server.

However, Fazli *et al* (2012) proposed a hardware-based authentication scheme to mitigate MitM and MitB attacks. Their objective is to provide trust communication between a client and a server as well as preserving the secrecy of the user's sensitive information. They incorporated **TPM(Trust Platform Module)** based remote attestation in order to provide the platform integrity verification. In addition, they adopted the **Secure Remote Password (SRP)** as the secure key exchange protocol in order to provide zero knowledge proof that allows one party to prove themselves to another without revealing any authentication information such as password.

Nilsson (2012) describes the security research for a web application designed by **BehavioSec**. The application uses JavaScript to record keystrokes to generate data that is sent back to the server for verification. On the client side the data is gathered from the users typing pattern and finally the data is transmitted to the server where it is validated, extracted and processed. The time spent is also evaluated and stored in the database which shows the behavioural techniques that he incorporated in order to make it difficult for the attacker.

3. METHODOLOGY

In this section, this paper presents a proposed solution to mitigate the MitM/MitB attacks. The goal of our technique is to make it difficult for the attacker to "grab" sensitive information exchange between client and server with the use of Anti-Form Grabbing technique and also to enforce more security on the verification aspect by introducing E-mail verification service for information exchange between the Client and Server. In order to achieve the objective for Anti-Form Grabbing, it will be done in such a way that all the data input typed in by the user will be automatically encoded making what is displayed on the screen different from what is being typed. When the user logs in to the financial website, a dynamic JavaScript will be generated from the server side which will now be sent to the client side. This means for every client that logs onto the server side, a unique JavaScript code is generated and sent to the client which manages the user inputs. Encoding users' individual JavaScript is monitored through their session ID. The concept behind the dynamic JavaScript is that any character that the user inputs from the keyboard or onscreen keypad is not the same character that is being displayed on the screen. The JavaScript is like a simple algorithm that encodes input of user and replaces it with another character/symbol on the screen, so whatever the user types is not the same thing that will be displayed in the form, so even if the attacker "grabs" the form input of the user, he has only "grabbed" combination of meaningless symbols which he can't decode. So when the user finishes with the form and submits to the server side, the server knows the meaning of what has been sent to base on the session ID of the client, since the server side was the one that generated the JavaScript for that session. It will then decode the input based on the dynamic JavaScript it sent to the client. In this way, every user logged on to a server has a unique session ID, so it will use your session ID to decode what JavaScript it sent to you in order to decode the user's input.

Also for the second objective which is the Email verification service, a new and secure way of verifying the authenticity of the user by rerouting the verification from the server through email verification service will be introduced. This means when the server wants to verify the user of the transaction, it will send the verification through another session instead of sending it through the same session on the browser. With this, the user will have to open an email through another session to verify the transaction. Since the Trojan infecting the browser is only triggered when the user is about to start a financial transaction, it will be difficult for the attacker to manipulate the information sent from the server with this method.

3.1 The Proposed Enhanced Security System

In this section, the modeling strategy employed in achieving the proposed enhanced security system will be considered. First, the system architecture and the flow diagram chart or the workflow shall be considered. Predominantly, during the implementation of the work, the workability of the security features will be actualized, but in this section, the theoretical functionalities of the various components of the enhanced security system will be considered. This contains two separate solutions which are discussed below. One showing how the data would be gathered and transmitted in a login session and one showing how the transaction could be verified. PHP will be used as server side scripting language and JavaScript will be used for the client side scripting. Apache will be used as web server software. A MySQL database server will also be used on the server machine.

Each of these technologies listed above have their respective part they played in realizing the proposed security measure for online financial web applications.

3.2 System Architecture

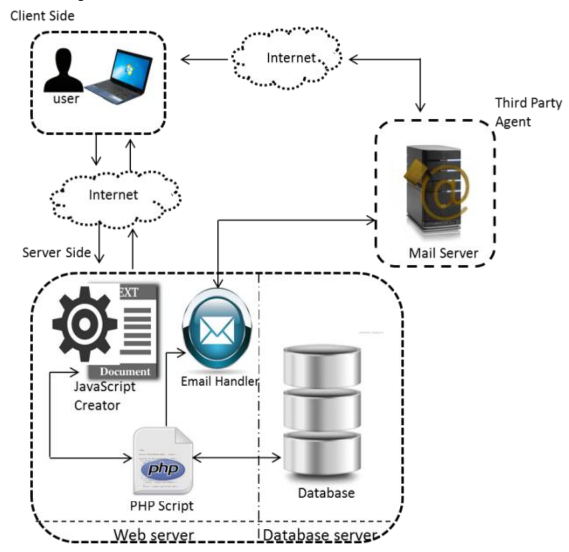


Figure 1: The System Architecture

a) Client Side: This is where the transaction is initiated. The user logs in his details and sends it to the server for authentication. It provides the interface for the user to be able to communicate with the server side like data inputs, submission of forms and so on.

I) User: The user is the person that makes use of the client side and initiates the transaction.

b) Server Side: The server side comprises of three components, which consists of the PHP script engine, JavaScript creator and the email handler. This is the power house of the system which connects all the three. It is the driving force of all the three components.

I) JavaScript Creator: This generates the dynamic JavaScript that will be sent to the client side which manages the user input. It creates a JavaScript code based on the session value of the user who has logged in. After creating the JavaScript, it will be sent to the client side along with the form. When it is sent to the client side, the user fills the form and sends it back.

II) PHP Script: These server side scripting are used to generate the web pages which analyses the data when transmitted to the server and also communicates with MySQL Server. It automatically engineers the creation of the JavaScript.

III) Email Handler: This is the aspect that handles email. Both the mail that is sent to the user and the confirmation mail from the user are handled by this component.

IV) Database: All the information is being stored on the database.

c) Third Party Agent: This is like a middle-man between the server side and the client side. It's the one that receives the mail from the server side and delivers the mail to the client side and vice versa. The server side sends the authentication mail to the user's email and he checks it, confirms it and sends it back to the server to confirm authentication. This is actually known as the mail server. This could include both commercial mail servers and private mail servers, though depending on the email address submitted by the user during registration with the financial house. This email server acts as a middle man in enforcing our proposed security measure.

3.3 System Flow Chart

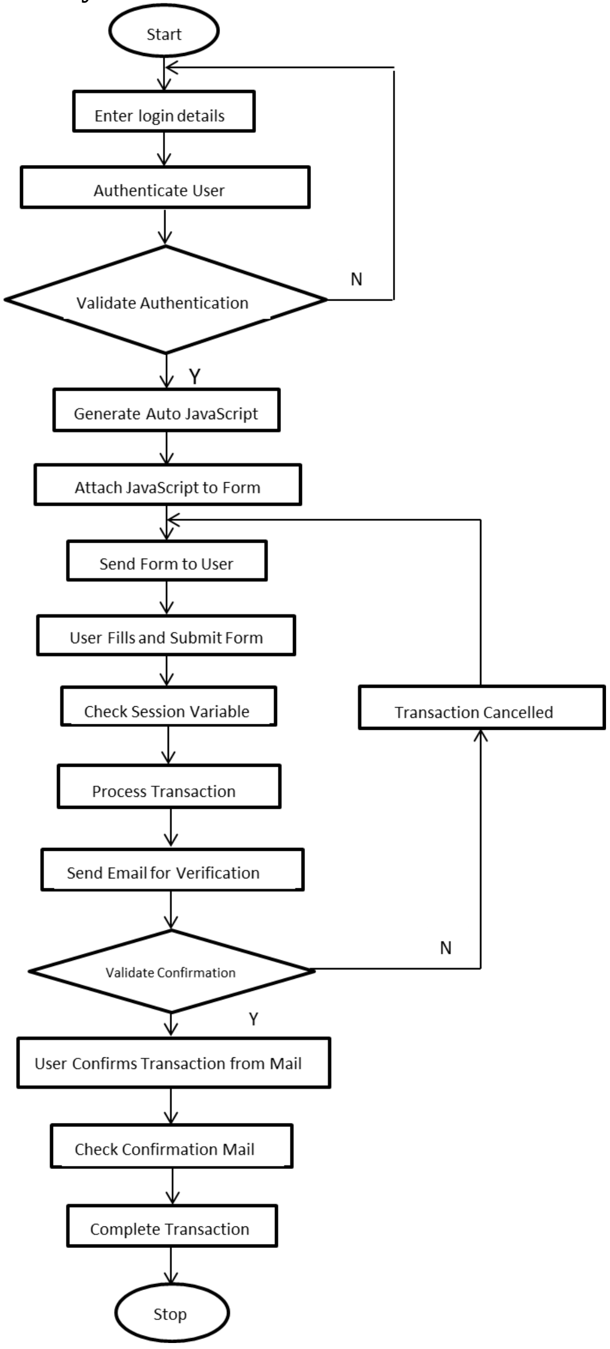


Figure 2: The Flow Chart

As shown above, Figure 2 captures the flow of information in the system. As soon as the user gets to the system, he must login with his login details and wait for authentication. The server will check whether the authentication is valid and if it is, it will now generate JavaScript and after generating it, it will now attach the JavaScript generated to the form that will be sent to the user. It is in that form that the user will fill all the transaction details and send it back to the server. The server checks for session variable in order to decode what has been filled in the form, then process the transaction and sends email to the user for verification. After sending the email for verification, the user opens his email box and confirms it because the transaction won't be complete unless the user confirms the transaction. If the user doesn't confirm the transaction either by not taking any action in time or he clicked on "decline", the transaction will be cancelled and it will return back to the user filling the form and submitting it again. But if the user confirms the transaction by clicking on "accept", then the transaction has completed successfully. So between the generation of the JavaScript and the user filling and submitting the form is where the Anti-Form Grabbing technique comes in. The generated JavaScript that is attached to the form is what makes the data that the user types in the form be encoded in such a way that is not readable before the form is sent to the server. So if the attacker "grabs" the form, he ends up stealing meaningless information.

3.4 The Security Model Mitigating MITB

In this session, the technique employed in mitigating Man-In-The-Browser attack through the Anti-Form Grabbing Technique will be considered and also how the Email Verification Service is going to be used.

3.4.1 The Anti-Form Grabbing Technique

This is the first of what is to be done to mitigate MITB on the financial transaction platform. Sensitive information which contains data fields that are labelled, Account Number and the Amount being entered will be encrypted. Whatever the user fills in the form will be sent to the user's email and when he opens his email, he confirms the transaction and sends it back to the server which now verifies the transaction. In this technique, a random generation number system with level randomization scheme is needed, an encryption algorithm specifically SHA512, PHP session variable and a pool of multilingual characters to make the system work. Once the user logs in, his login details in transit to and from the server will be encrypted automatically to thicken the security measure. After the user has logged in successfully and he is authenticated, the PHP session variable value for that specific user is attached to that user which will now be stored at the server side. The specialize random number generation algorithm will now be used to generate 62 characters (A-Z, a-z, 0-9) from a pool of 100 characters for each user. The random numbers will be generated from 1to 62. For example, assuming we generate a character pool of 19, from the multilingual characters that are there, it will pick a character at position 19 in the pool of our overall multilingual. So if a user logs in, the 62 characters generated randomly will be attached to that user and for another user that logs in, also another set of 62 characters will be generated for its transactional purpose. The idea behind this different alphabets system among different users of the same system is to make it more difficult for hackers who may want to exploit security weakness through MITB to compromise the transaction. Here, since every user has a continual changing alphabetical system, it will be difficult for a sniffer to observe the alphabetical system with a notion of compromising the system in subsequent attack. Also, even if the attacker successfully hijacks transaction data in transit, what it would have captured will be a sequence of multilingual character that may not make any reasonable meaning except it is decoded. The encoding for the alphabet system of a given user is done using the random number scheme described above and embedded in dynamically generated JavaScript. So this shows if two or more users are accessing the website at the same time, their number systems are different. So even if the attacker grabs the users' forms, he will notice that they are different number systems, so he can't make meanings out of their details filled in the form which is why it is called Anti-Form Grabbing Technique. So the attacker is confused over a range of users since each user has a different number system. Even if the attacker tries the attack another period for the same user, the number system will be different from the other time he earlier attacked. So with this format, the multi-lingual alphabet system will be generated and sent to the user along with the form but it will be encoded in form of JavaScript.

Algorithm 1: Algorithm for Anti-Form Grabbing Technique

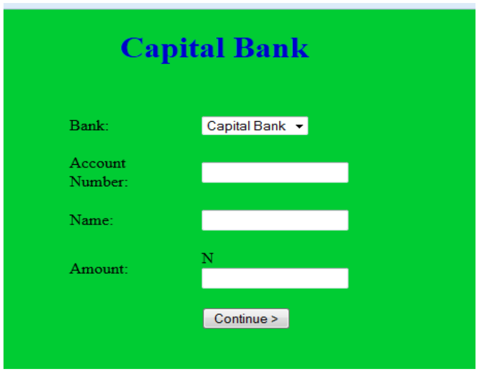
```
1    Read character pool $P = \text{Ƒ}_{(alp)} + \text{Ƈ}_{(alp)} + \text{Ɠ}_{(alp)}$
2    Get session ID
3    let Web App alphabet system be $\text{Ɛ}_{(Alp)} \equiv [A\ldots Z, a\ldots z, 0\ldots 9]$
4    for $i$ range from 1 to 62
         $\mu(p)_i = P[\text{fRand}(.)]$
         where $\mu(p)_1 \ldots \mu(p)_{62} \in \text{Ɛ}_{(Alp)}$
5    foreach $\mu(p)_i$ mapped into $\text{Ɛ}_{(Alp)i}$
6    Create dynamic JavaScript.
     |    • Encode all $\mu(p)_i$ into the JavaScript.
     |    • Denote each $\mu(p)_i$ with its corresponding $\text{Ɛ}_{(Alp)i}$ on the client side.
     |    • Tag dynamic JavaScript with user session ID.
     end loop
7    Send transaction form with dynamic JavaScript.
8    if user sends form back
     |    then
     |        |    i.    Decode user inputs by mapping all $\mu(p)_i$ back to $\text{Ɛ}_{(Alp)i}$
     |        |    ii.   Send verification email
     |        |    iii.  if email is confirmed, go to step 9
     |    else wait
     |
9    |Complete transaction based on user
     agree/disagree option through mail
10   end transaction.
```

3.4.2 Email Verification Service

This is the second aspect which will buttress the security of the first one in order to have a firm security. After the Anti-Form Grabbing Technique has done its job, an email is sent from the email handler in the server to the user's email box to verify the transaction whether it is valid or not. This is as indicated in Algorithm 1. This is relatively out of the reach of the attacker since he cannot monitor both the transaction and the email verification option at the same time. Moreover, note that user is expected to respond to email alert within a given time, else the transaction will be rendered invalid. Two URL (Universal Resource Locator) links will be sent to the clients email in order to confirm whether the transaction is valid or not. If it is valid and correct, the user will click the link that agrees with the transaction details sent to the mail. Else the user can decline the transaction by clicking on the "**disagree**" URL link.

4. RESULTS AND DISCUSSION

In this paper, the following result is shown below:

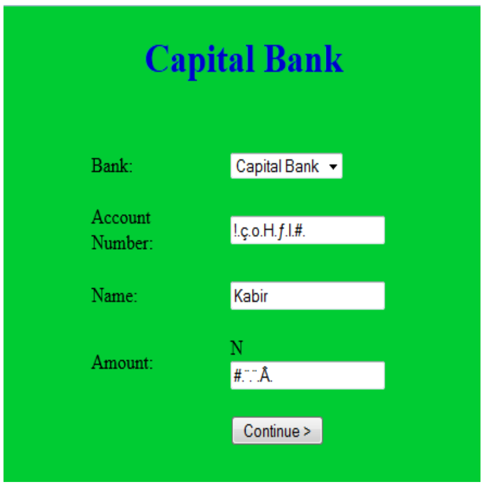


Capital Bank

Bank:	Capital Bank
Account Number:	
Name:	
Amount:	N

Figure 3: The Login Screenshot

The Login screenshot in Figure 3 displays when the client logs in to the financial website. During this time, a JavaScript is being generated from the server which is now sent to the client side.

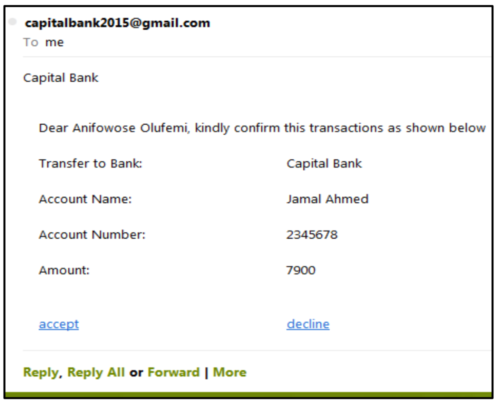


Capital Bank

Bank:	Capital Bank
Account Number:	!.ç.o.H.ƒ.l.#.
Name:	Kabir
Amount:	N #.¨.¨.Â.

Figure 4: The Form Screenshot

The Form Screenshot in Figure 4 shows the form being filled by the client. As the client fills the transaction details especially in the "Account Number" and the "Amount" section, automatically the keystrokes are being converted and replaced with symbols unknown to the attacker which is where Anti-Form grabbing takes place.



capitalbank2015@gmail.com
To me

Capital Bank

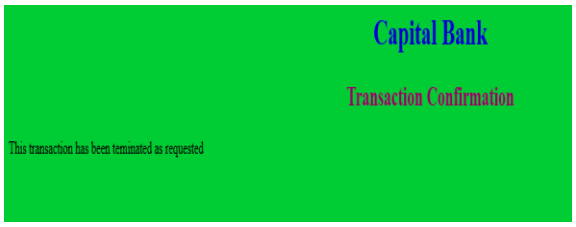
Dear Anifowose Olufemi, kindly confirm this transactions as shown below

Transfer to Bank:	Capital Bank
Account Name:	Jamal Ahmed
Account Number:	2345678
Amount:	7900
accept	decline

Reply, Reply All or Forward | More

Figure 5: Email Screenshot

The Screenshot in Figure 6 is the email sent to the client's email box after the submission of the client's form. The client checks his email to verify if the information is correct or not. Based on the information he sees, he is expected to either click "accept" or "decline".



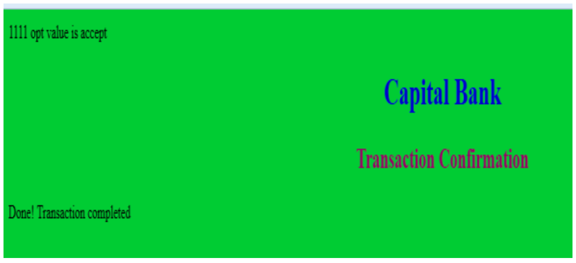
Capital Bank

Transaction Confirmation

This transaction has been teminated as requested

Figure 6: Transaction Termination Screenshot

The Screenshot in Figure 6 is the Transaction Termination. This is displayed when the client clicks on "decline" in order to stop the transaction.



1111 opt value is accept

Capital Bank

Transaction Confirmation

Done! Transaction completed

Figure 7: Accepted Transaction Screenshot

The Screenshot in Figure 7 is the Transaction Accepted. This is displayed when the client clicks on "accept" in order for the transaction to go through if he is satisfied with the information he has seen. This means the transaction is completed successfully.

5. CONCLUSION

In this paper, an enhanced solution to the security of web applications that are specific to online transactions was proposed. The information being exchanged between the client and the server will be protected by introducing Anti-form grabbing in which the user inputs are immediately captured and encoded by a dynamically generated JavaScript from the server side. To enforce more security on the verification aspect, E-mail verification service was introduced for information exchange between the Client and Server. With this, MITB attacks can be prevented and if not, then at least make it more difficult for the attacker to succeed with an attack.

6. FUTURE WORKS

In order to test the proposed security model, it was implemented against intra-banking transaction scenario, so a future research should consider an inter-banking scenario so that the security model will be able to cover for the prospective client's other financial institution outside the security model.

7. REFERENCES

[1] Abbasi, A.G., Muftic, S., and Hotamov, I. (2010). *Web Contents Protection, Secure Execution and Authorized Distribution*, Computing in the Global Information Technology, Fifth International Multi-conference on Computing in the Global Information Technology, International Multi-Conference on, pp. 157-162.

[2] Akinwale, T. A., Adekoya, F. A., and Ooju, E. O. (2011). *Multi-Level Cryptographic Functions for the Functionalities of Open Database System*, Department of Computer Science, University of Agriculture, Abeokuta, Nigeria.

[3] Association of German Banks. (2007). *Online banking security*. Berlin: Bundesverband deutsher Banken.

[4] Batchelor, B., *The History of E-Banking*. Retrieved August 11 2014 from http://www.ehow.com/about_5109945_history-ebanking.html

[5] Boswell, W. (2014)., *The History of the Web*. Retrieved August 10 2014 from http://websearch.about.com/od/searchingtheweb/a/webhistory.htm

[6] Canali, D., and Balzarotti, D. (2013). *Behind the Scenes of Online Attacks: an Analysis of Exploitation Behaviors on the Web*. NDSS 2013, 20th Annual Network and Distributed System Security Symposium, February 24-27, 2013, San Diego, CA, United States.

[7] Fazli, B., Kamarularifin, A., and Jamalul-lail, A. (2012). International Journal of Cyber-Security and Digital Forensics (IJCSDF). *Mitigating Man-In-The-Browser Attacks with Hardware-based Authentication Scheme*. 1(3): 204-210.

[8] Nilsson, D. (2012). *Security in Behaviour Driven Authentication for Web Applications*, Master thesis, Department of Computer Science, Electrical and Space Engineering.

[9] Jason, W., Damien, H., and Justin, P. (2008). *Enhanced Security for Preventing Man-in-the-Middle Attacks in Authentication, Data Entry and Transaction Verification.* Deakin University: Australian Information Security Management Conference.

[10] Jjchai.(2010)., *Online banking*. Retrieved August 11 2014 from http://www.slideshare.net/jjchai/online-banking

[11] RSA Lab, *Making Sense of Man-in-the-browser Attacks*, http://viewer.media.bitpipe.com/1039183786_34/1295277188_16/MITB_WP_05 10-RSA.pdf.

[12] Scholasticus, K.(2009)., *History of Internet Banking*. Retrieved August 11 2014 from http://www.buzzle.com/articles/history-of-internet-banking.html

[13] Sidheeq, M., Dehghantanha, A., and Kananparan, G. (2010). *Utilizing trusted platform module to mitigate botnet attacks*, Computer Applications and Industrial Electronics, International Conference on, vol., no., pp. 245-249.

[14] Weigold, W., Kramp, T., Hermann, R., Horing, F., Buhler, P., and Baentsch, M. (2008). *The Zurich Trusted Information Channel: An efficient defense against Man-in-the-middle and malicious software attacks* TRUST'2008. LNCS, vol. 4968, pp. 75-91.